Techno.COM, Vol. 11, No. 2, Mei 2012: 89-96

RANCANG BANGUN PROTOTYPE RMI (*REMOTE METHOD INVOCATION*) UNTUK MENGHUBUNGKAN SISTEM BANK JATENG DENGAN SISTEM PEMBAYARAN UDINUS

Ajib Susanto[1)], Mochammad Lukman[2)]
[1,2)]Program Studi Teknik Informatika, Fakultas Ilmu Komputer
Universitas Dian Nuswantoro Semarang
Jl. Nakula I No. 5-11 Semarang 50131
Telp : (024) 3517261, Fax : (024) 3520165
E-mail : [1)]ajibsusanto@gmail.com, [2)]lmnn_blidibi@yahoo.com

Abstrak

Perkembangan teknologi internet memungkinkan banyak device dapat saling terhubung. Mekanisme untuk memisahkan atau mendistribusikan data dan aplikasi agar bisa di akses di jaringan menjadi sangat penting. Java adalah salah satu pemrograman yang sangat mensupport teknologi ini dengan menyediakan RMI (Remote Method Invocation), dimana dengan RMI dapat tercipta suatu sistem yang terdistribusi karena RMI membolehkan sebuah obyek yang berada dalam sebuah JVM untuk meng-invoke method dari obyek lain yang berada dalam JVM yang terpisah. Dengan kata lain, RMI memungkinkan untuk mengirim obyek sebagai parameter dari remote method. Ada beberapa cara untuk membangun suatu sistem yang terdistribusi menggunakan RMI yaitu: mendefinisikan remote interface, implementasi remote interface dan server, pengembangan client yang menggunakan remote interface, mengkompilasi source files dan membuat stub and skeletons, memulai (start) RMI registry, dan menjalankan server dan client.

Kata kunci : *RMI, JVM, sistem bank.*

Abstract

Development of Internet technology allows many devices can be connected to each other. Mechanism for separating or distributing data and applications that can be accessed on the network becomes very important. Java is a programming technology that greatly supports this by providing the RMI (Remote Method Invocation), where by RMI to create a distributed system because the RMI allows an object to be in a JVM to invoke a method of another object within the separate JVM. In other words, RMI allows to send the object as a parameter of the remote method. There are several ways to build a distributed system using RMI are: defining the remote interface, remote interface and implementation of server, client development using the remote interface, compile the source files and create stubs and skeletons, start (start) RMI registry, and running the server and client.

Keywords : *RMI, JVM, banking system.*

1. PENDAHULUAN

Informasi sudah sejak lama memegang peran penting dalam kehidupan karena hampir semua keputusan penting selalu dihasilkan berdasarkan informasi yang terbaru. "Sumber kekuatan baru bukanlah uang yang berada dalam genggaman tangan beberapa orang, namun informasi di tangan banyak orang,"[1]. Dalam perkembangan teknologi di era globalisasi ini informasi dapat dengan mudah didapatkan melalui banyak media, tetapi terkadang informasi tidak bisa didapatkan karena sistem yang menyediakan informasi ini tidak membolehkan untuk mengakses informasi tersebut.

Dalam perkembanganya, suatu sistem diklasifikasikan menjadi beberapa bagian, salah satunya yaitu sistem terbuka (*Open System*) dan sistem tertutup (*Closed System*). Sistem terbuka adalah sistem yang berhubungan dan terpengaruh dengan lingkungan luarnya. Sistem ini menerima masukan dan menghasilkan keluaran untuk lingkungan luar atau subsistem yang lainnya. Sedangkan sistem tertutup adalah sistem yang tidak berhubungan dan tidak terpengaruh dengan lingkungan luarnya. Sistem ini bekerja secara otomatis tanpa adanya turut campur tangan dari pihak diluarnya. Akan tetapi kenyataannya tidak ada sistem yang benar-benar tertutup, yang ada hanyalah *relatively closed system*.

Dunia perbankan dan dunia pendidikan adalah dunia yang sama sekali berbeda, yang tentunya system yang digunakan pasti berbeda. Namun dua dunia tersebut saling membutuhkan, salah satu contohnya adalah dimana dunia pendidikan membutuhkan dunia perbankan untuk mengorganisir aktivitas keuanganya, dan dunia perbankan membutuhkan dunia pendidikan untuk pengembangan dunia perbankan itu sendiri.

Sistem pembayaran adalah salah satu bagian dari sistem yang digunakan oleh semua instansi, baik instansi pemerintah ataupun instansi swasta. UDINUS (Universitas Dian Nuswantoro) adalah salah satu instansi swasta yang bergerak dalam bidang pendidikan, tentunya juga menggunakan sistem pembayaran yang sesuai dengan kebutuhan instansi ini. Jika dilihat dalam perkembangannya UDINUS telah bekerjasama dengan beberapa Bank untuk mengkoordinir pembayaran baik untuk kebutuhan mahasiswa ataupun dosen.

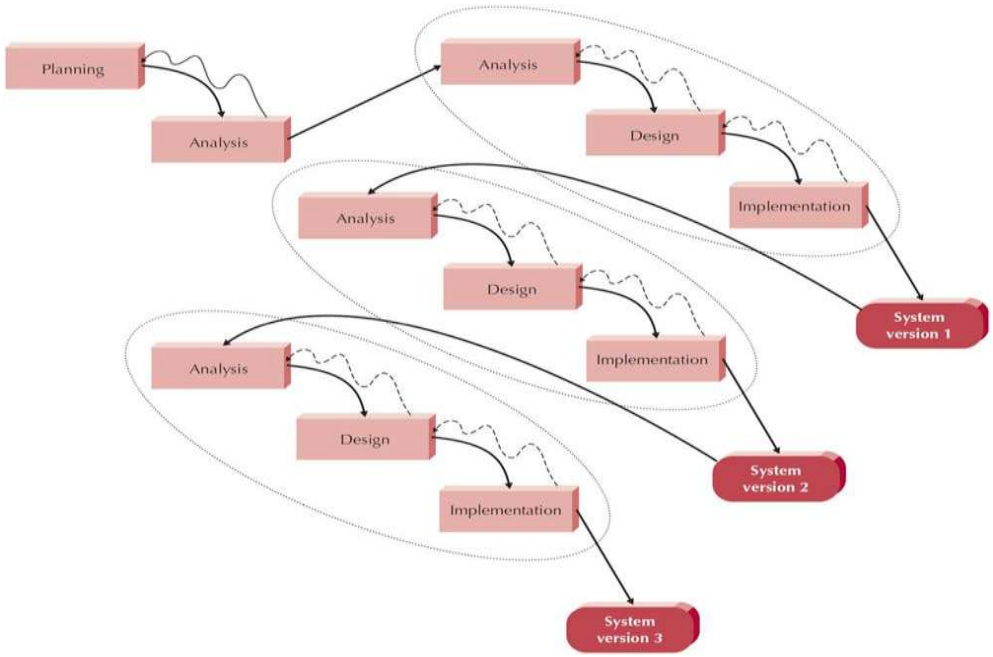
Namun walaupun kelihatanya sudah bekerjasama dengan pihak Bank, informasi pembayaran sering kali terlambat diterima baik terlambat dari pihak Bank, maupun dari pihak UDINUS. Ini dikarenakan masing-masing instansi ini memiliki sistem yang berbeda sedangkan mereka sama-sama menerapkan sistem tertutup sebagai parameter sistem mereka untuk alasan khusus. Sehingga walaupun mereka bekerjasama, pengiriman informasi pembayaran masih manual karena masing-masing sistem tidak terhubung secara langsung [2][3].

Pertumbuhan jumlah mahasiswa yang bernaung dalam instansi ini yang semakin bertambah setiap tahunnya, tentu jika model sistem ini tetap dilanjutkan, akan sangat membuang waktu mengingat semua keputusan yang diberikan terkadang juga sangat bergantung pada informasi pembayaran. Hal ini sering menghambat aktivitas mahasiswa yang ingin melanjutkan kegiatan perkuliahanya, karena sistem menghentikan prosesnya jika belum ada informasi pembayaran yang masuk yang diakibatkan oleh terlambatnya informasi pembayaran diterima oleh pihak UDINUS.

Dari Permasalahan-permasalahan tersebut, dibutuhkan adanya suatu sistem yang dapat mengakomodasi banyak sistem yang tertutup menjadi dapat saling bertukar informasi.

2. METODE

Rancang bangun sistem ini menggunakan pengembangan perangkat lunak dengan model *prototype*. Model tersebut dipilih karena Dengan metode prototyping ini pengembang dan *user* dapat saling berinteraksi selama proses pembuatan system. *User* dapat menunjukkan fitur yang disukai dan tidak disukai, mengindikasikan apa yang diinginkan pada sistem yang sudah ada dan berjalan lebih mudah dari pada harus mendeskripsikannya secara teoritis. Pengalaman dan penggunaan lebih menghasilkan informasi penting dari pada diagram analisis dan proposal naratif [4][5][6].



Gambar 1. *Konsep Perkembangan Prototyping*

Tahapan-tahapan dalam Prototyping adalah sebagai berikut:

1. Pengumpulan kebutuhan
   Pelanggan dan pengembang bersama-sama mendefinisikan format seluruh perangkat lunak, mengidentifikasikan semua kebutuhan, dan garis besar sistem yang akan dibuat.
2. Membangun prototyping
   Membangun prototyping dengan membuat perancangan sementara yang berfokus pada penyajian kepada pelanggan (misalnya dengan membuat input dan format output)
3. Evaluasi protoptyping
   Evaluasi ini dilakukan oleh pelanggan apakah prototyping yang sudah dibangun sudah sesuai dengan keinginann pelanggan. Jika sudah sesuai maka langkah 4 akan diambil. Jika tidak prototyping direvisi dengan mengulangu langkah 1, 2 , dan 3.
4. Mengkodekan sistem
   Dalam tahap ini prototyping yang sudah di sepakati diterjemahkan ke dalam bahasa pemrograman yang sesuai
5. Menguji sistem
   Setelah sistem sudah menjadi suatu perangkat lunak yang siap pakai, harus dites dahulu sebelum digunakan. Pengujian ini dilakukan dengan White Box, Black Box, Basis Path, pengujian arsitektur dan lain-lain
6. Evaluasi Sistem
   Pelanggan mengevaluasi apakah sistem yang sudah jadi sudah sesuai dengan yang diharapkan . Jika ya, langkah 7 dilakukan; jika tidak, ulangi langkah 4 dan 5.
7. Menggunakan sistem
   Perangkat lunak yang telah diuji dan diterima pelanggan siap untuk digunakan .

3. PEMBAHASAN

3.1. Analisa Aplikasi

Tahap analisa Aplikasi adalah studi domain masalah untuk merekomendasikan perbaikan dan menspesifikasi persyaratan dan prioritas untuk solusi. Tugas paling penting dalam tahap ini adalah proses menemukan masalah dan menghasilkan alternatif pemecahan masalah serta diharapkan dapat memahami aplikasi yang ada guna menentukan kebutuhan pemakai dan hambatan pada aplikasi yang baru.

Identifikasi Masalah

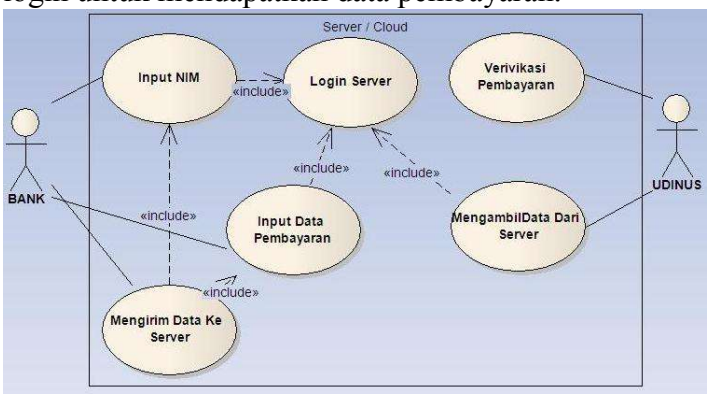
Permasalahan yang terjadi adalah ketika mahasiswa selesai membayarkan tagihan nya di Bank, mahasiswa tidak dapat langsung melihat hasilnya pada sistem yang sudah ada, dikarenakan Bank baru mengirimkan data pembayaran pada periode waktu tertentu dan ditambah data yang dikirim adalah data printout yang pada akhirnya pihak Universitas akan menginputkan lagi datanya. Hal ini mengakibatkan menjadi lamanya proses verifikasi pembayaran, sehingga pada kasus-kasus tertentu, download kartu ujian misalnya, kejadian seperti ini akan sangat mengganggu sekali.

Pemecahan Masalah

Perlu adanya suatu aplikasi yang menjembatani antara sistem keuangan UDINUS yang tertutup, dan sistem informasi keuangan Bank yang tertutup agar dapat saling bertukar informasi secara real time demi menunjang keefektifitasan proses pembayaran dan verifikasinya tanpa menggangu sistem yang sudah berjalan pada masing-masing pihak.

3.2. Perancangan

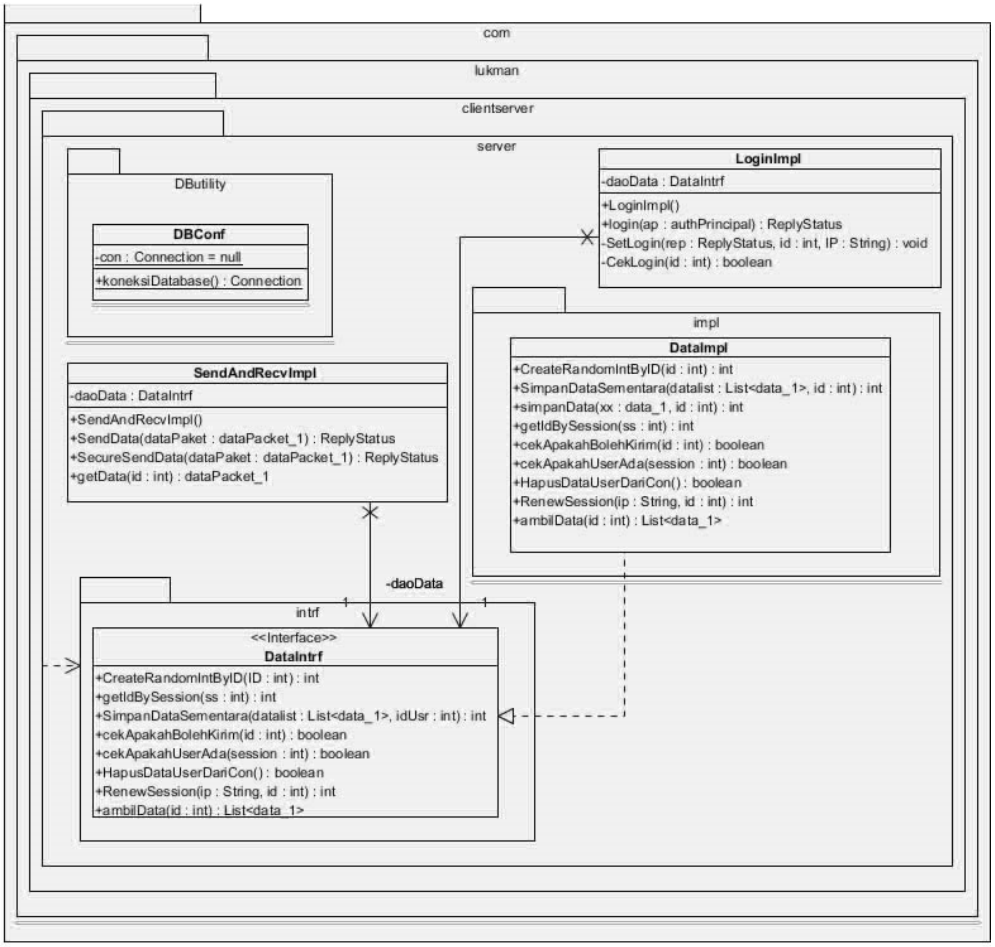
Use Case Diagram untuk menggambarkan interaksi antara actor dan sistem di Bank Jateng dan Udinus. Untuk dapat mengirimkan data pembayaran pihak yang mengirimkan data harus login ke server. Pihak penerima data tidak perlu melakukan login untuk mendapatkan data pembayaran.



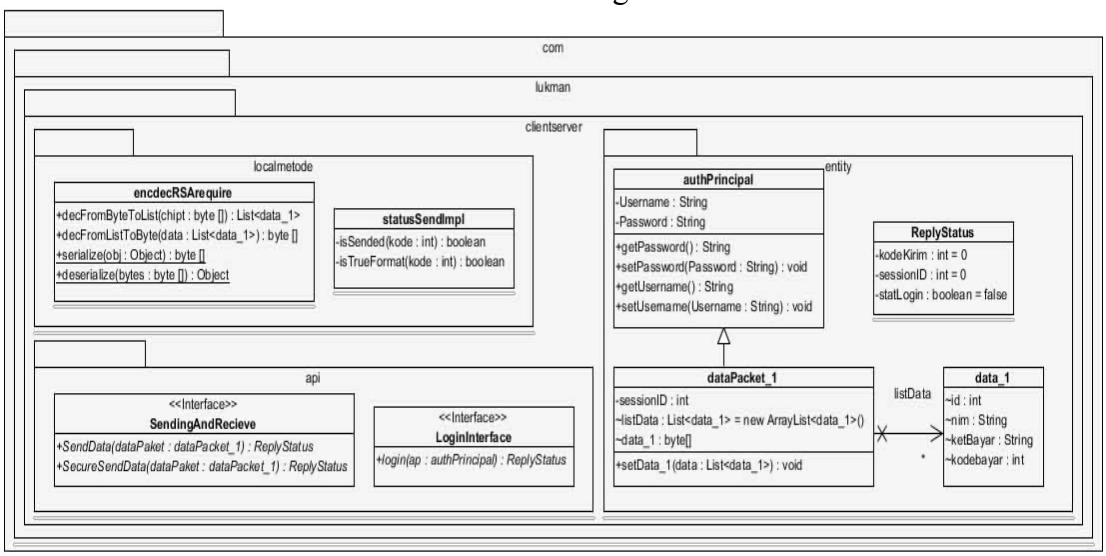
Gambar 2. *Use Case Diagram*

Class diagram

Class diagram menggambarkan hubungan antar *class* dan hubungan suatu *class* dengan *class* pendukungnya. Gambar diagram *class* dapat dilihat pada gambar :

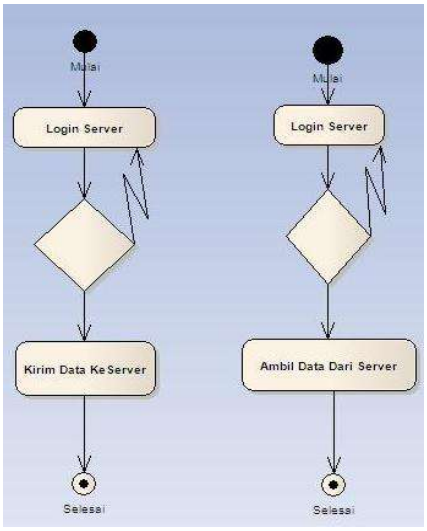


Gambar 3. Class Diagram Server



Gambar 4. Class Diagram API

Diagram Aktivitas



Gambar 5. Proses Pengiriman Data Pengambilan Data

Dalam pengiriman maupun pengambilan data dari atau ke server, masing-masing tetap ada yang meng interupsi untuk kasus-kasus tertentu.

Gambar 6. Proses pengaktivan server

Agar layanan ini dapat berjalan harus ada trigger yang mengaktifkan server.

Perancangan Database

Perancangan database dibutuhkan dalam pengembangan sistem ini sebagai struktur dasar dari database dan tabel yang akan digunakan untuk menyimpan data yang nantinya akan diakses oleh aplikasi tersebut. Rancangan struktur database dan tabel yang digunakan sebagai berikut :

Tabel User

Field	Type
id_user	int(11)
Ip	varchar(15)
Uname	varchar(32)
Pass	varchar(32)
Port	int(4)

Tabel User Detail

Field	Type
id_detail	int(11)
id_user	int(11)
nama	varchar(30)
alamat	text
perusahaan	text

Tabel RSA

Field	Type
id_user	int(11)
Pubkey	Longblob
Privkey	Longblob
Id	int(11)

Tabel Connection

Field	Type
id_login	int(11)
id_user	int(11)
ip_address	varchar(15)
time_last_connect	Datetime
session	int(11)

Tabel Log Connection

Field	Type
id_log	int(11)
time_last_connect	Datetime
Session	int(11)

Tabel Data Sementara

Field	Type
id	int(11)
nim	varchar(20)
jenis	varchar(30)

3.3 Implementasi Sistem

User Interface

User interface design merupakan salah satu aspek yang penting dalam pembangunan sebuah sistem. *User interface design* harus dibuat sesuai dengan tujuan aplikasi, target pasar yang dituju, bersifat *user friendly* serta menarik sehingga dapat menumbuhkan minat *user* untuk mencoba untuk menggunakan aplikasi tersebut. Berikut *user interface design Prototype RMI* :

Gambar 7. *Control Server Ketika Servis Non Aktif*

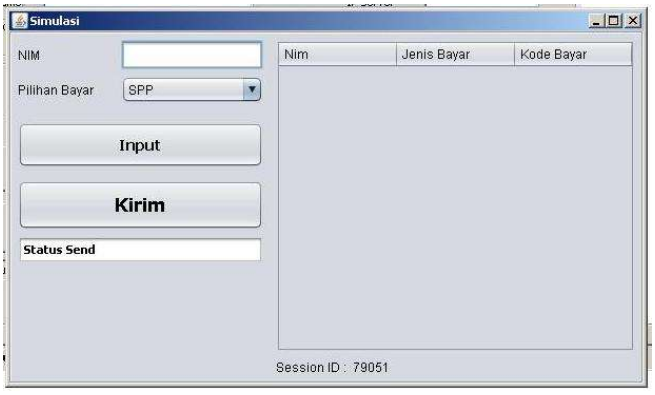
Pada Form ini menunjukkan bahwa servis pada server pada kondisi tidak aktif. Dengan menekan tombol aktifkan, maka servis yang disediakan server akan dijalankan.

Gambar 8. *Control Server Ketika Servis Aktif*

Pada form ini menunjukkan bahwa servis pada server dalam keadaan aktif, informasi yang didapatkan adalah waktu dimana servis aktif, waktu *real time*, durasi servis aktif, status aktif server dan aktif *registry* yang dibuat *server*. Jika ditekan tombol matikan, maka *server* akan berhenti menjalankan servisnya dan status maupun tampilan akan kembali menjadi seperti saat *server* tidak aktif.

Gambar 9. Form Simulasi Pengambilan Data

Untuk dapat mengambil data dalam server, cukup mengisikan data alamat server dan port yang digunakan layanan tersebut yang kemudian disinkronkan dengan API yang tersedia.



Gambar 4.10. *Form Simulasi Pengiriman Data*

4. Simpulan

Kesimpulan yang diambil dari hasil implementasi adalah:

a. Tidak ada sistem yang benar-benar tertutup, yang ada hanya setengah tertutup atau *relatively closed system*.
b. Efisiensi dalam penggunaan waktu yang dibutuhkan untuk mengolah data lebih cepat dan menghasilkan informasi yang lebih akurat dan lebih cepat.

5. DAFTAR PUSTAKA

[1] Pitt, Esmond. (2006). *Fundamental Networking In Java*.
[2] Bima, Ifnu. (2010). *Java Desktop.*
[3] Riyanto, Zaki. 2004. *Komunikasi Data.*
[4] Pressman, Roger S. (2005). *SOFTWARE ENGINEERING: A Practitioner's Approach (terjemahan*). Yogyakarta: Penerbit Andi
[5] http://java.sun.com/developer/ onlineTraining/rmi / RMI.html, diakses tanggal 12 Februari 2012
[6] http: // docs.oracle.com/ javase/ tutorial/ networking/index.html, diakses tanggal 25 Januari 2012